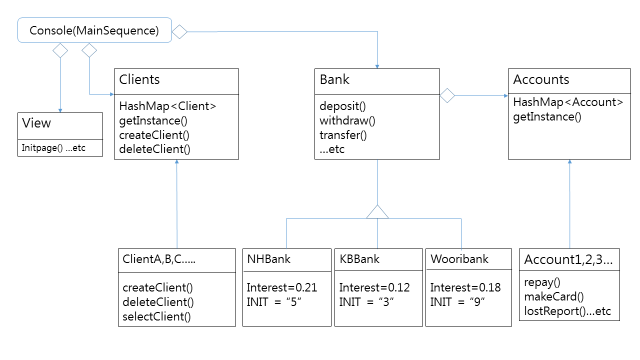
banKMU

콘솔에서 Client의 요청을 받아 Bank의 Account에 접근해 입/출금 등 은행의 기능들을 수행하는 프로그램

Eclipse, junit 이용

[Class diagram]



클래스 관계

Bank는 은행, Client는 고객, Account는 Bank에서 관리하는 Client의 계좌이다.  
인스턴스 생성, 메서드 실행, 사용자의 명령은 모두 MainSequence에서 제어한다.  
MainSequence는 콘솔에 매뉴얼을 출력하기 위한 View인스턴스, Client를 다루기 위해 Clients 인스턴스, Client의 Account에 접근하기 위한 Bank 인스턴스를 가진다.

인스턴스 관리

Clients는 Client를 관리하기 위해, Accounts는 Account를 관리하기 위해 해쉬맵을 이용한다.  
Client의 해쉬키는 곧 Client 생성순서이다. Accounts의 해쉬키는 각 Bank의 INIT값이 키의 시작문자가 되며 같은 공유 풀을 이용하는 해쉬맵에서 서로 다른 은행에서 만든 키 값이 겹칠 경우는 없다. 키값의 나머지 5자리는 랜덤하게 정해진다.

Bank를 상속받은 세 은행은 필요에 의해 생성되고 삭제된다. 콘솔에서 특정 은행을 선택하면 해당 은행의 인스턴스가 MainSequence의 멤버변수에 저장된다.

콘솔 인터페이스

자바 콘솔로 구현되며 매뉴얼에 따라 몇 개의 화면으로 구성된다.  
최상위 화면은 Client 선택창이며 새로운 Client를 생성하는 것도 가능하다.

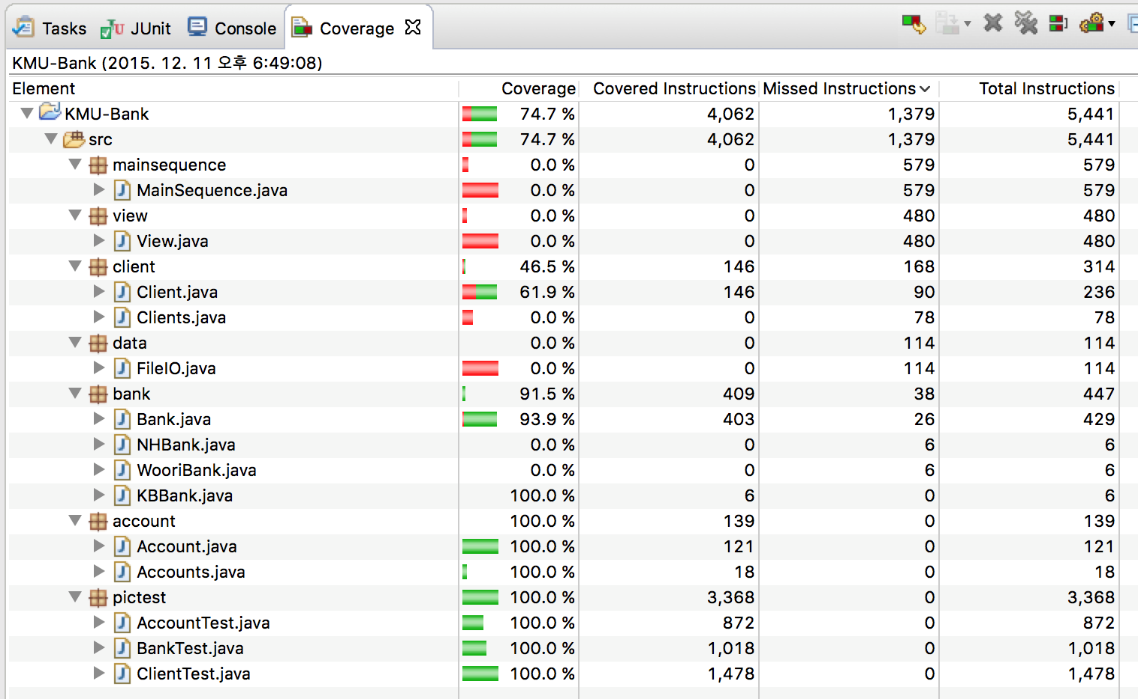
생성된 Client를 선택하면 이미 존재하는 세 개의 은행 중 어떤 은행을 선택할지 결정할 수 있다. 은행을 선택하면 은행에서 제공하는 기능이 나열된 화면이 등장한다.

이때 Client에게 해당 은행의 Account가 없다면 Account를 생성할 것인지 결정하는 화면이 등장한다. 이미 Account가 존재한다면 해당 은행에서 제공받을 수 있는 기능들(입,출금,송금 등)이 등장한다.

Testing

테스팅은 pairwise를 이용하며 Pairwise를 위한 도구로 pict를 사용한다.

테스팅은 크게 TestBank, TestClient, TestAccount 세 개의 테스트 클래스로 구성된다.  
pict결과로 나온 테스트케이스들은 각각 하나의 Junit 테스팅 함수로 작성된다.



>> 최종 code coverage : 74.7%